Problem B. Find The Car

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 секунда
Memory limit:	512 мебибайт

Это интерактивная задача.

К концу года на работе у Штирлица был полный завал. Так что, когда он вышел из здания Рейхсканцелярии, то увидел, что все стоящие у здания n машин находятся под снегом.

Штирлиц не помнит, где именно он поставил машину, но он знает, что машины сотрудников Рейхсканцелярии всегда выстроены «почти по порядку». То есть или машины выстроены слева направо по возрастанию номеров, либо существует такое $0 < k < n$, что номера первых слева k машин идут по возрастанию, затем ровно один раз порядок нарушается, затем номера снова идут по возрастанию, при этом номер самой правой машины оказывается меньше номера самой левой.

Штирлиц может за минуту откопать машину, чтобы посмотреть её номер. Сумеет ли он найти свою машину за полчаса? Номера всех машин являются попарно различными целыми положительными числами, не обязательно идущими подряд.

Interaction protocol

На стандартный поток ввода задаются два числа n и s — общее количество машин у здания и номер машины Штирлица ($1 \le n \le 10^6; 1 \le s \le 10^9$).

После этого Ваша программа должна вывести число i ($1 \le i \le N$) — номер слева очередной машины, которую раскапывает Штирлиц. После вывода числа не забудьте вывести перевод строки, а также сбросить буфер вывода функцией `flush`.

В ответ программа получает на стандартный поток ввода номер c_i ($1 \le c_i \le 10^9$) откопанной машины. Гарантируется, что числа расположены в соответствии с условием задачи, что число s присутствует среди этих чисел и что в процессе проверки порядок машин не меняется (то есть интерактор является статическим).

После того, как в ответ получено число s, Ваша программа должна завершить выполнение. Напоминаем, что если количество запросов будет превышать 30, то Ваше решение не будет зачтено.

Examples

standard input	standard output
7 20 2 15 20	4 1 2

Problem C. Dialogue

Input file:	*standart input*
Output file:	*standart output*
Time limit:	1 секунда
Memory limit:	512 мебибайт

Это интерактивная задача.

Во время межзвёздной экспедиции n космонавтов-исследователей вышли в открытый космос для проведения экспериментов в ваккууме. Учёные получили скафандры с номерами от 1 до n.

К сожалению, во время выдачи скафандров случился сбой и соответствие «номер скафандра — имя использующего его учёного» было утрачено. Система резервирования данных позволяет делать запросы следующего типа: задаётся некоторое количество различных целых чисел от 1 до n, после чего система выводит список имён учёных, использующих эти скафандры, в **случайном** порядке.

Требуется восстановить соответствие не более, чем за 50 запросов (если задано более 50 вопросов, система резервирования считает это метеоритной DDoS-атакой и переходит в режим самозащиты).

Input

На ввод программе подается одно целое число n — количество проводящих эксперименты в ваккууме учёных, $(1 \le n \le 512)$.

Далее следуют ответы на запросы.

Если был сделан некорректный запрос или превышено максимальное количество запросов, то ответом на запрос будет ‘‘`-1`’’ (без кавычек). После получения такого ответа ваша программа должна немедленно завершиться.

Если был сделан запрос на получение имен, то в первой строке ответа будет содержаться одно целое число m — количество названных имен. Следующие m строк будут содержать сами имена в случайном порядке. Имена состоят только из строчных латинских букв. Длина каждого имени не превосходит 20.

Гарантируется, что в составе экспедиции нет двух учёных с одинаковыми именами.

Output

Запрос выглядит следующим образом.

Если вы спрашиваете имена, то в первой строке запроса выведите ‘‘`?`’’ (без кавычек), во второй строке выведите количество запрашиваемых имен. В третьей строке выведите номера скафандров, для которых вы хотите узнать имена. Нельзя спрашивать имя одного человека дважды в одном запросе.

Если вы хотите вывести ответ, то в первой строке запроса выведите ‘‘`!`’’ (без кавычек), в следующей строке выведите ‘‘`-1`’’ (без кавычек), если ответ получить нельзя. Иначе, выведите список имен по возрастанию номеров соответствующих им скафандров.

После вывода ответа программа должна немедленно завершиться.

Не забывайте сбрасывать буфер вывода после вывода каждого запроса. Это делается при помощи функции ‘‘`fflush(stdout)`’’; в ‘‘`C++`’’, ‘‘`System.out.flush()`’’ в ‘‘`Java`’’ и ‘‘`flush(output)`’’; в ‘‘`Pascal`’’.

Имя одного человека можно спрашивать несколько раз, но не в одном запросе. Можно делать полностью одинаковые запросы.

Examples

standart input	standart output
3 3 caba aba rab 2 rab aba 2 aba caba 1 aba 1 aba 1 caba 1 rab	? 3 1 2 3 ? 2 1 3 ? 2 1 2 ? 1 1 ? 1 1 ? 1 2 ? 1 3 ! aba caba rab

Problem D. Intelligence test

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 секунды
Memory limit:	512 мебибайт

Одна из частей теста IQ, принятого в Байтландии, состоит в следующем: необходимо вычеркнуть из заданной последовательности несколько чисел определённым образом.

Ваша задача — написать программу автоматической проверки того, что некоторая последовательность может быть получена из заданной путём вычёркивания одного или нескольких чисел.

Input

Первая строка входного файла содержит целое число m ($1 \le m \le 1\,000\,000$) — длину первоначальной последовательности. В следующей строке заданы m целых чисел $a_1, a_2, \ldots, a_m$ ($1 \le a_i \le 1\,000\,000$ для $1 \le i \le m$) — элементы первоначальной последовательности. В третьей строке записано целое число n — количество последовательностей, вложение которых нужно проверить. Последующие $2n$ строк содержат описания этих последовательностей. Каждая последовательность описывается двумя строками. Первая из них содержит целое число m_i ($1 \le m_i \le 1\,000\,000$) — длину последовательности. Вторая содержит последовательность из m_i элементов: $b_{i,1}, b_{i,2}, \ldots, b_{i,m_i}$ ($1 \le b_{i,j} \le 1\,000\,000$ для $1 \le j \le m_i$). Суммарная длина всех n последовательностей не превосходит 1 000 000.

Output

Для каждой из n последовательностей выведите в отдельной строке в порядке следования во входном файле слово“TAK” (*да* по-польски) если соответствующая последовательность может быть получена вычёркиванием нескольких чисел из первоначальной, или “NIE” (*нет* по-польски) в противном случае.

Examples

standard input	standard output
7 1 5 4 5 7 8 6 4 5 1 5 5 8 6 3 2 2 2 3 5 7 8 4 1 5 7 4	TAK NIE TAK NIE

Problem E. Frog

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 секунды
Memory limit:	512 мебибайт

На дне длинного прямого ручья лежат n камней, выступающих над поверхностью воды.

Их расстояния от истока ручья равны $p_1 < p_2 < \ldots < p_n$ соответственно. Лягушка прыгает с камня на камень следующим образом. Каждый раз она прыгает на камень, который является k-м ближайшим к тому, на котором она сидит. Например, если лягушка сидит на камне с координатой p_i, p_j определяется следующим образом:

$$|\{p_a \ : \ |p_a - p_i| < |p_j - p_i|\}| \le k \qquad \text{and} \qquad |\{p_a \ : \ |p_a - p_i| \le |p_j - p_i|\}| > k.$$

Если p_j можно выбрать несколькими способами, выбирается камень, ближайший к истоку ручья.

По заданному исходному расположению лягушки вычислите, на каком камне она будет сидеть после m прыжков.

Input

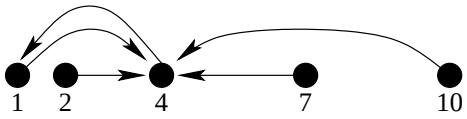
В первой строке входного файла содержатся три целых числа n, k и m ($1 \le k < n \le 1\,000\,000$, $1 \le m \le 10^{18}$), — количество камней, параметр k и количество предполагаемых прыжков. Вторая строка содержит n целых чисел p_j ($1 \le p_1 < p_2 < \ldots < p_n \le 10^{18}$) — расстояния от истока ручья до соответствующих камней.

Output

В единственной строке выходного файла выведите n чисел. i-е число обозначает номер (начиная с 1) камня, на котором лягушка будет сидеть после m прыжков, начиная с i-го камня.

Examples

standard input	standard output
5 2 4 1 2 4 7 10	1 1 3 1 1



На иллюстрации показано, куда лягушка прыгнет с каждого камня.

Problem F. Disciples and Discipline

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 секунды
Memory limit:	512 мебибайт

Профессор Северус был назначен руководителем отделения алхимии зимней магической школы для одарённых детей. Профессор решил, что для повышения управляемости лучшим решением будет ввести строгую иерархию, создав тайный орден алхимиков среди школьников и преподавателей.

Всего в работе отделения участвует N человек. Каждый из них имеет номер временного пропуска в здание школы от 1 до N. Пропуск с номером 1 и звание магистра ордена, разумеется, имеет сам Северус. У каждого из участников школы с номером от 2 до N есть непосредственный мастер, причём ровно один. Если участник x является мастером для участника y или существует цепочка из $k > 0$ участников x_i такая, что x является мастером для участника x_1, участник x_i является мастером для участника x_{i+1} для всех i от 1 до $k-1$, а участник x_k является мастером для участника y, считается, что y находится под ответственностью x. При этом гарантируется, что ни один участник не находится под ответственностью самого себя. Все участники, под ответственностью которых находится хотя бы один участник, являются преподавателями, остальные являются школьниками.

Узким местом при организации школы являются очереди в столовую. Северус решил определить порядок, в котором школьники будут посещать столовую. Однако при определении порядка надо учесть M пожеланий от руководства школы. Требования имеют формат «школьник на a_i-м месте в очереди должен иметь номер пропуска b_i (если соответствующий номер пропуска выдан школьнику) или находиться под ответственностью преподавателя с номером пропуска b_i (если соответствующий номер пропуска выдан преподавателю)».

Северусу стало интересно, сколько существует различных способов определить порядок посещения столовой школьниками, удовлетворив всем требованиям руководства школы. Два способа считаются различными, если порядок прохода в столовую отличается хотя бы в одном месте. Поскольку ответ может быть очень большим, выведите результат по модулю $10^9 + 7$.

Input

Первая строка входного файла содержит два целых числа N и M ($2 \le N \le 3 \times 10^5, 0 \le M \le 3 \times 10^5$) — количество участников отделения алхимии и количество пожеланий руководства, соответственно.

Номер пропуска 1 имеет сам Северус. В каждой i-й строке из последующих $N - 1$ строк находится единственное число $1 \le p_i < i + 1$ — номер пропуска мастера для участника с номером пропуска $i + 1$. В каждой i-й строке из последующих M строк заданы пожелания; каждое пожелание состоит из двух чисел $1 \le a_i, b_i \le 3 \times 10^5$ — a_i-й школьник в очереди должен либо иметь номер пропуска b_i, либо находиться под ответственностью преподавателя с номером пропуска b_i. Гарантируется, что a_i не превосходит количества школьников отделения алхимии.

Output

В выходной файл выведите единственное число — остаток от деления количества различных способов составить расписание посещения столовой на $10^9 + 7$.

Examples

standard input	standard output
5 3 1 1 2 2 1 2 2 1 3 2	2
6 1 1 1 2 2 2 4 3	6

Problem G. Sheep

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 секунды
Memory limit:	512 мебибайт

Пастбища в Байтландии представляют собой выпуклые многоугольники. При этом у каждой из овец на пастбище есть «своя точка» строго внутри (не на сторонах) пастбища — место, где она проводит практически всё время. Также иногда овцы устраивают поединки друг с другом. Байтландские пастухи считают, что это полезно для укрепления навыков самозащиты от хищников, поэтому в стаде каждого пастуха количество овец чётно, чтобы каждая овца могла найти себе соперника для поединка.

В одном байтландском селе в соответствии с рекомендациями специалистов решили провести следующий эксперимент: пастбище разделили на треугольники с помощью $n-3$ внутренних перегородок, соединяющих вершины многоугольника и не пересекающихся нигде, кроме этих вершин. При этом ни одна из перегородок не должна проходить в точности через место, в котором обычно находится какая-то овца, и в каждом из получившихся треугольников должно быть чётное количество «своих точек», чтобы сохранить возможность поединков.

Требуется вычислить количество подобных разбиений первоначального пастбища.

Input

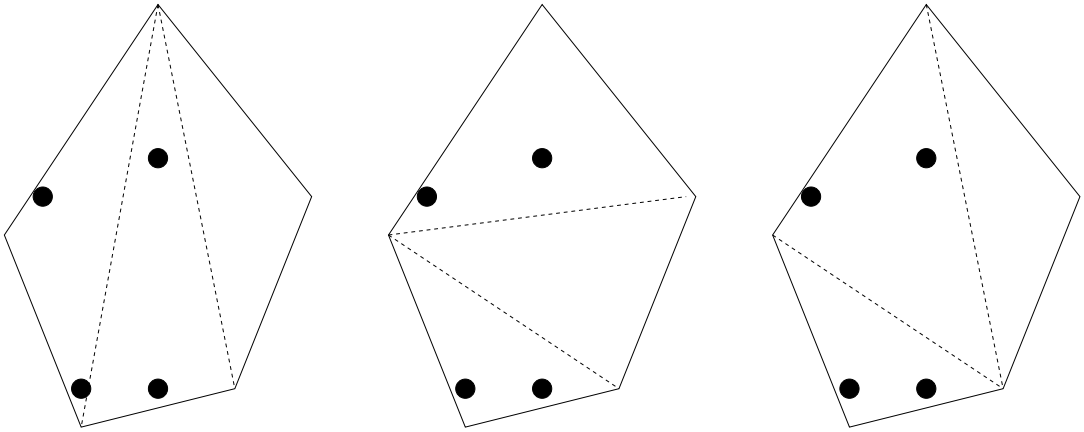
Первая строка входного файла содержит три целых числа n, k и m ($4 \le n \le 600$, $2 \le k \le 20\,000$, $2 \mid k$, $2 \le m \le 20\,000$) — количество вершин многоугольника, образующего пастбище, количество овец и модуль m, который потребуется для вывода ответа. Каждая из последующих n строк содержит два целых числа x_i и y_i ($-15\,000 \le x_i, y_i \le 15\,000$) — координаты i-й вершины пастбища. Вершины заданы в порядке обхода по часовой стрелке. Каждая из последующих k строк содержит два целых числа p_j, q_j ($-15\,000 \le p_j, q_j \le 15\,000$) — координаты «своей точки» для j-й овцы.

Output

Выведите одно целое число — остаток от деления количества описанных в условии разбиений на m.

Examples

standard input	standard output
5 4 10 5 5 3 0 -1 -1 -3 4 1 10 1 0 -1 0 1 6 -2 5	3



На иллюстрации приведены все три разбиения пастбища на треугольники. Также на карте обозначены «свои точки» всех овец.